# TIER API POC on Grouper demo server

TIER Authz demo on grouper demo server (use chrome recommended).  User/pass:  email the list to get access

- Has member
  - https://grouperdemo.internet2.edu/grouper-ws_v2_2/tierApiAuthz/v1/Groups/name:test:testGroup/Members/id:test?indent=true
- Doesn't have member
  - https://grouperdemo.internet2.edu/grouper-ws_v2_2/tierApiAuthz/v1/Groups/name:test:testGroup/Members/id:mchyzer@upenn.edu?indent=true

## Setup

Generic TIER authz server:

https://github.com/Internet2/grouper/tree/GROUPER_2_2_BRANCH/grouper-misc/tierApiAuthzServer

Grouper implementation of a back end:

https://github.com/Internet2/grouper/tree/GROUPER_2_2_BRANCH/grouper-misc/grouper-tierApiAuthz-connector

Build those to jars, put it in grouper 2.2

Put in the config files:

https://github.com/Internet2/grouper/tree/GROUPER_2_2_BRANCH/grouper-misc/grouper-tierApiAuthz-connector/src/resources

Add  tierApiAuthz.server.properties:

```
# Server type in the service meta.  e.g. Grouper WS 2.1.23
tierApiAuthzServer.serverType = Grouper WS 2.2.2


tierApiAuthzServer.servletUrl = http://localhost:8089/grouperWs/tierApiAuthz


# groups member logic, implement edu.internet2.middleware.tierApiAuthzServer.interfaces.
AsasApiGroupsMemberInterface
tierApiAuthzServer.interface.groupsMember = edu.internet2.middleware.grouperTierApiAuth.interfaces.
GtaasGroupsMemberInterfaceImpl


########################################
## Client configuration
########################################
# dir where dirs can hold client configs.  Files could be in this dir or in a subdir
tierApiAuthzServer.clientConfigDir = /opt/tomcats/tomcat_i/clients
```

https://github.com/Internet2/grouper/tree/GROUPER_2_2_BRANCH/grouper-misc/tierApiAuthzServer/src/resources

Add grouperTierApiAuthz.server.properties which is blank

Add this to the web.xml

```
  <filter>
    <filter-name>TIER API authz server filter</filter-name>
    <filter-class>edu.internet2.middleware.tierApiAuthzServer.j2ee.TaasFilterJ2ee</filter-class>
  </filter>
  <!-- Map the filter to a Servlet or URL -->
  <filter-mapping>
    <filter-name>TIER API authz server filter</filter-name>
    <url-pattern>/tierApiAuthz/*</url-pattern>
  </filter-mapping>
  <servlet>
    <servlet-name>TierApiAuthzServlet</servlet-name>
    <display-name>TIER API authz Servlet</display-name>
    <servlet-class>edu.internet2.middleware.tierApiAuthzServer.j2ee.TaasRestServlet</servlet-class>
    <load-on-startup>1</load-on-startup>
  </servlet>
  <servlet-mapping>
    <servlet-name>TierApiAuthzServlet</servlet-name>
    <url-pattern>/tierApiAuthz/*</url-pattern>
  </servlet-mapping>

<!-- optional if you need this, configure appropriately -->
  <security-constraint>
    <web-resource-collection>
      <web-resource-name>Web services</web-resource-name>
      <url-pattern>/tierApiAuthz/*</url-pattern>
    </web-resource-collection>
    <auth-constraint>
      <role-name>grouper_user</role-name>
    </auth-constraint>
  </security-constraint>
```

## Client configuration

Each client can have its own configuration to define which fields to return etc

Set the directory where the client configs are in tierApiAuthz.server.properties

```
########################################
## Client configuration
########################################


# dir where dirs can hold client configs.  Files could be in this dir or in a subdir
tierApiAuthzServer.clientConfigDir = C:\\Users\\mchyzer\\Documents\\GitHub\\grouper_v2_2\\grouper-
misc\\tierApiAuthzServer\\clients
```

Have a base config e.g. clients/common/baseClient.properties

```
# properties shared among all clients

# The tier api server uses Configuration Overlays (documented on wiki)
# By default the configuration is read from authzStandardApi.server.base.properties
# (which should not be edited), and the authzStandardApi.server.properties overlays
# the base settings.  See the tierApiAuthz.server.base.properties for the possible
# settings that can be applied to the tierApiAuthz.server.properties
########################################
## Config chaining hierarchy
########################################


# comma separated config files that override each other (files on the right override the left)
# each should start with file: or classpath:
# e.g. classpath:someFile.server.example.properties, file:c:/something/myconfig.properties
#tierClient.config.hierarchy = classpath:someFile.server.base.properties, classpath:someFile.server.properties
# this needs to be in each client config file
# seconds between checking to see if the config files are updated
#tierClient.config.secondsBetweenUpdateChecks = 60


########################################
## Client config
########################################


# set this for a non helper config, which specifies which users it is applicable for
# tierClient.users =
# show exception stack in response
tierClient.showExceptionStack = false


########################################
## Generic operations
########################################


# show name in get group member
tierClient.generic.showName = true


########################################
## Get group member operation
########################################


# show name in get group member: true, false, or inherit from tierClient.generic.showName
tierClient.getGroupMember.showName = inherit
```

You can have generic overrides for all clients, e.g. clients/common/baseClientCustom.properties

```
########################################
## Client config
########################################


# show exception stack in response
tierClient.showExceptionStack = false
```

Each client can have its own config or you can share among clients, e.g. clients/GrouperSystem/GrouperSystem.properties

```
# comma separated config files that override each other (files on the right override the left)
# each should start with file: or classpath:
# e.g. classpath:someFile.server.example.properties, file:c:/something/myconfig.properties
tierClient.config.hierarchy = file:C:\\Users\\mchyzer\\Documents\\GitHub\\grouper_v2_2\\grouper-
misc\\tierApiAuthzServer\\clients\\common\\baseClient.properties, file:C:
\\Users\\mchyzer\\Documents\\GitHub\\grouper_v2_2\\grouper-
misc\\tierApiAuthzServer\\clients\\common\\baseClientCustom.properties, file:C:
\\Users\\mchyzer\\Documents\\GitHub\\grouper_v2_2\\grouper-
misc\\tierApiAuthzServer\\clients\\GrouperSystem\\GrouperSystem.properties
# this needs to be in each client config file
# seconds between checking to see if the config files are updated
tierClient.config.secondsBetweenUpdateChecks = 60



########################################
## Client config
########################################



# set this for a non helper config, which specifies which users it is applicable for, comma separated
tierClient.users = GrouperSystem
```

## Logging

Add this to log4j.properties to get request logging

```
log4j.appender.tierApiAuthz_requestLog                          = org.apache.log4j.DailyRollingFileAppender
log4j.appender.tierApiAuthz_requestLog.File                     = /some/folder/tierApiAuthz_request.log
log4j.appender.tierApiAuthz_requestLog.DatePattern              = '.'yyyy-MM-dd
log4j.appender.tierApiAuthz_requestLog.layout                   = org.apache.log4j.PatternLayout
log4j.appender.tierApiAuthz_requestLog.layout.ConversionPattern  = %d{ISO8601}: %m%n

log4j.logger.edu.internet2.middleware.tierApiAuthzServer.logging.TaasRequestLog = DEBUG, tierApiAuthz_requestLog
log4j.additivity.edu.internet2.middleware.tierApiAuthzServer.logging.TaasRequestLog=false
```